WHAT IS CLAIMED IS:

1. A CAN microcontroller that supports a plurality of message objects, comprising:

a processor core that runs CAN applications;

a CAN/CAL module that processes incoming messages;

data memory including a first memory segment that provides a plurality of message buffers associated with respective ones of the message objects, and a second memory segment that provides a plurality of memory-mapped registers for each of the message objects, the memory-mapped registers for each message object containing respective command/control fields for configuration and setup of that message object; and,

a memory interface unit that permits the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory segments, and that arbitrates access to the same one of the first and second memory segments when the processor core and the CAN/CAL module request concurrent access to the same one of the first and second memory segments.

2. The CAN microcontroller as set forth in Claim 1, wherein the incoming messages include multi-frame, fragmented messages, and the CAN/CAL module automatically assembles the multi-frame, fragmented messages.

3. The CAN microcontroller as set forth in Claim 1, wherein the CAN/CAL module includes the memory-mapped registers.

4. The CAN microcontroller as set forth in Claim 1, wherein the processor core, the CAN/CAL module, and the memory interface unit are contained on a single integrated circuit chip.

5. The CAN microcontroller as set forth in Claim 4, wherein the first and second memory segments are contained on the integrated circuit chip.

6. The CAN microcontroller as set forth in Claim 4, wherein the memory interface unit includes two independent arbiters.

7. The CAN microcontroller as set forth in Claim 1, wherein the memory interface unit arbitrates access according to an alternate winner policy, wherein a previous loser is designated a current winner.

8. A CAN microcontroller that supports a plurality of message objects, comprising:

a processor core that runs CAN applications;

a CAN/CAL module that processes incoming messages, wherein the processor core and the CAN/CAL module are contained on a single integrated circuit chip;

data memory including a first memory space that is located on the integrated circuit chip and a second memory space that is located off the integrated circuit chip, the first memory space including a first memory segment that provides at least a portion of a message buffer memory space that includes a plurality of message buffers associated with respective ones of the message objects, and a second memory segment that provides a plurality of memory-mapped registers for each of the message objects, the memory-mapped registers for each message object containing respective command/control fields for configuration and setup of that message object; and,

a memory interface unit that permits the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory spaces, that permits the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory segments, and that arbitrates access to the second memory space and that arbitrates access to the same one of the first and second memory segments when the processor core and the CAN/CAL module request concurrent access to the second memory space or to the same one of the first and second memory segments.

9. The CAN microcontroller as set forth in Claim 8, wherein the incoming messages include multi-frame, fragmented messages, and the CAN/CAL module automatically assembles the multi-frame, fragmented messages.

10. The CAN microcontroller as set forth in Claim 8, wherein the CAN/CAL module includes the memory-mapped registers.

11. The CAN microcontroller as set forth in Claim 8, wherein the memory interface unit is contained on the single integrated circuit chip.

12. The CAN microcontroller as set forth in Claim 8, wherein the second memory space provides at least a portion of the message buffer memory space.

13. The CAN microcontroller as set forth in Claim 8, wherein the memory interface unit includes two independent arbiters dedicated to a respective one of the first and second memory spaces.

14. The CAN microcontroller as set forth in Claim 8, wherein the memory interface unit

arbitrates access according to an alternate winner policy, wherein a previous loser is designated a current winner.

15. A method for operating a CAN microcontroller that supports a plurality of message objects, the CAN microcontroller including a processor core that runs CAN applications, a CAN/CAL module that processes incoming messages, and a data memory including a first memory space that is located on the integrated circuit chip and a second memory space that is located off the integrated circuit chip, the first memory space including a first memory segment that provides at least a portion of a message buffer memory space that includes a plurality of message buffers associated with respective ones of the message objects, and a second memory segment that provides a plurality of memory-mapped registers for each of the message objects, the memory-mapped registers for each message object containing respective command/control fields for configuration and setup of that message object, the method comprising:

permitting the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory segments; and,

arbitrating access to the same one of the first and second memory segments when the processor core and the CAN/CAL module request concurrent access to the same one of the first and second memory segments.

16. The method as set forth in Claim 15, wherein the arbitrating access step is performed in accordance with an alternate winner policy, wherein a previous loser is designated a current winner.

17. The method as set forth in Claim 16, wherein the arbitrating step is performed by a memory interface unit contained in the CAN microcontroller.

18. A method for operating a CAN microcontroller that supports a plurality of message objects, the CAN microcontroller including a processor core that runs CAN applications, a CAN/CAL module that processes incoming messages, and a data memory including a first memory space that is located on an integrated circuit chip on which the CAN microcontroller and the CAN/CAL module are incorporated, and a second memory space that is located off the integrated circuit chip, the first memory space including a first memory segment that provides at least a portion of a message buffer memory space that includes a plurality of message buffers associated with respective ones of the message objects, and a second memory segment that provides a plurality of memory-mapped registers for each of the message objects, the memory-mapped registers for each message object containing respective command/control fields for

configuration and setup of that message object, the method comprising:

permitting the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory spaces;

permitting the processor core and the CAN/CAL module to concurrently access a different respective one of the first and second memory segments;

arbitrating access to the second memory space when the processor core and the CAN/CAL module request concurrent access to the second memory space; and,

arbitrating access to the same one of the first and second memory segments when the processor core and the CAN/CAL module request concurrent access to the first and second memory segments.

19. The method as set forth in Claim 18, wherein the arbitrating access step is performed in accordance with an alternate winner policy, wherein a previous loser is designated a current winner.

20. The method as set forth in Claim 19, wherein the arbitrating step is performed by a memory interface unit contained in the CAN microcontroller.